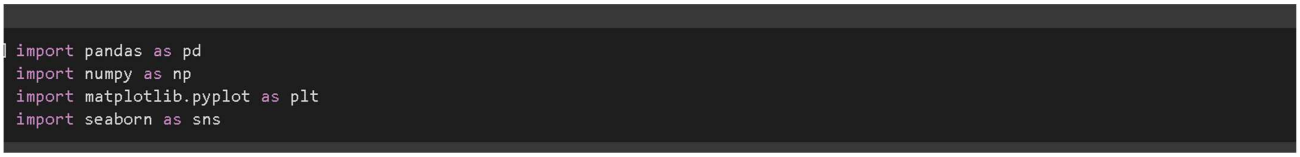
ML-MAJOR-MAY

We'll be using **Machine Learning** to identify digit classification using the **SVM Algorithm** using the **MNIST Dataset**.

In particularly, this report has following sections:

1. Overview
2. Data Description
3. Data Exploration
4. Data Preparation
5. Training and evaluating the machine learning model
6. Making predictions with model

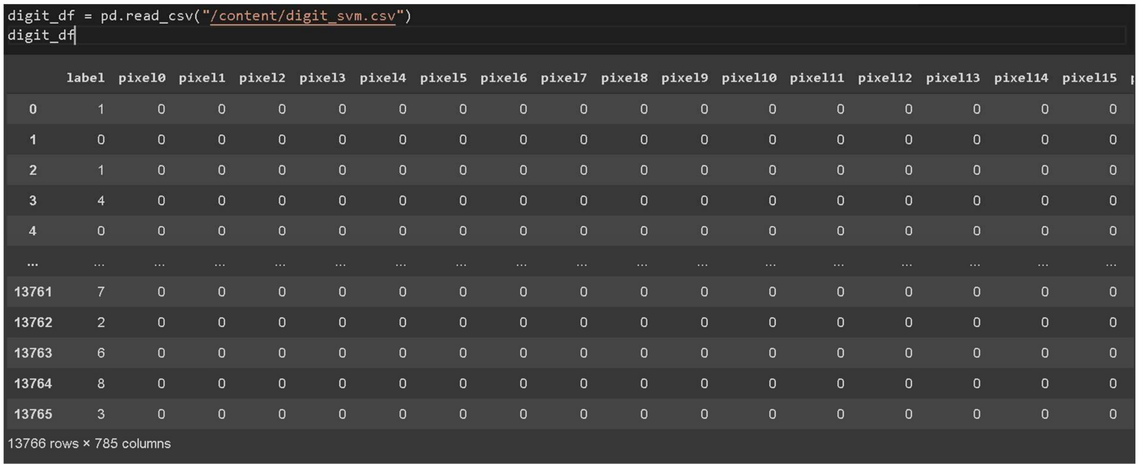
We'll be using **Python** and some of its popular data science related packages. First of all, we will import **pandas** to read our data from a CSV file and manipulate it for further use. We will also use **numpy** to convert out data into a format suitable to feed our model.



```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Data Description:

We have our data saved in a CSV file called **digit_svm.csv**. We first read our dataset into a pandas dataframe called digit_df. And then print the dataframe.

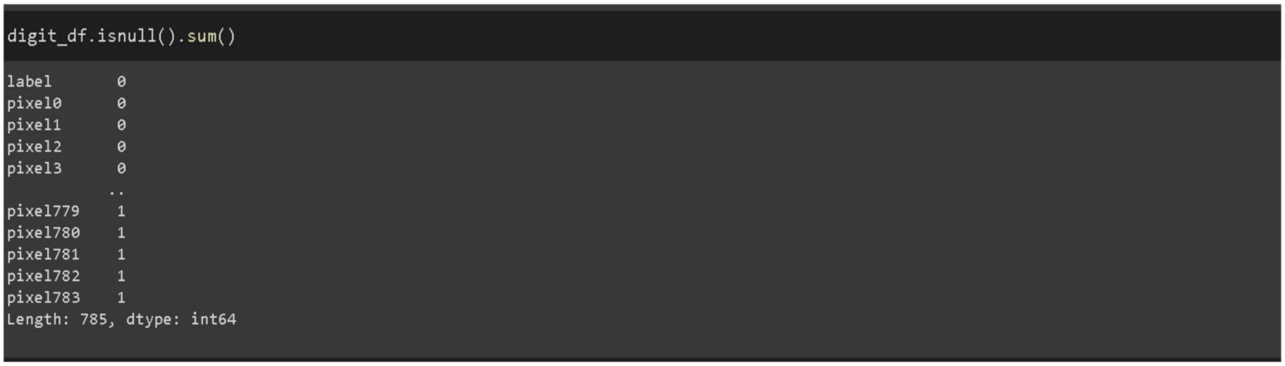


```
digit_df = pd.read_csv("/content/digit_svm.csv")
digit_df
```

	label	pixel0	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	pixel9	pixel10	pixel11	pixel12	pixel13	pixel14	pixel15
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
13761	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13762	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13763	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13764	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13765	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13766 rows × 785 columns

Let’s also make sure that our data is clean (has no null values, etc).



```
digit_df.isnull().sum()
```

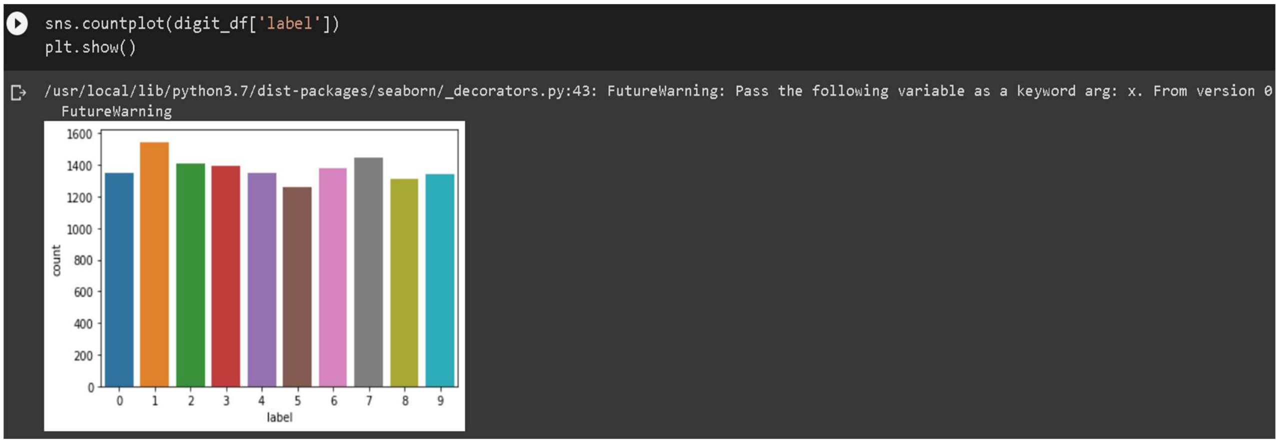
```
label       0
pixel0      0
pixel1      0
pixel2      0
pixel3      0
           ..
pixel779    1
pixel780    1
pixel781    1
pixel782    1
pixel783    1
Length: 785, dtype: int64
```

Since there are some NaN values in our dataset, we will replace them from 0 by using the following syntax:

digit_df=digit_df.fillna(0)

Data Exploration:

We can check whether the training data-set is biased towards certain numbers from the distribution plot of labels by using **seaborn** and **matplotlib**.



```
sns.countplot(digit_df['label'])
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0
  FutureWarning
```

Data preparation:

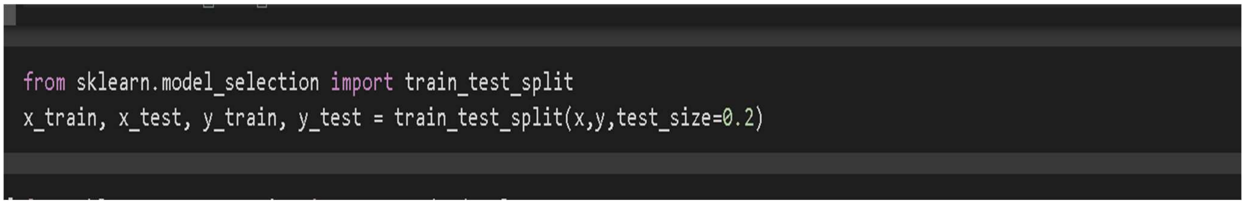
We will divide our data into input and output i.e. x and y using following syntax,

x = digit_df.iloc[:,1:]

y = digit_df.iloc[:, 0]

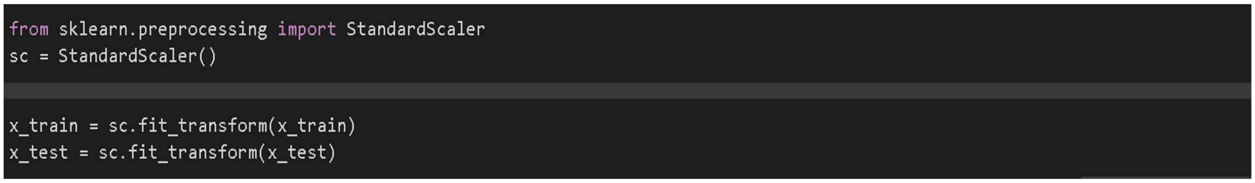
When using machine learning algorithms we should always split our data into a training set and test set. (If the number of experiments we are running is large, then we should be dividing our data into 2 parts, namely — training set and test set).

Our dataset consist of [13766 rows x 785 columns], To train our model we will be using **80%** of records and for testing we will be using **20%** of records.



```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.2)
```

Now we will the standardize the input, i.e. **x_train** and **x_test**, so that about 68% of values will lie in range **–1 to +1**.

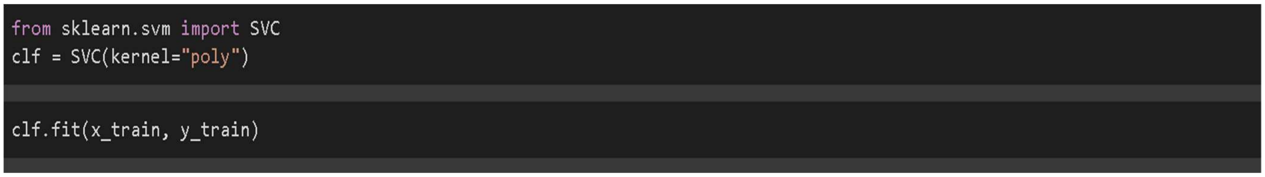


```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()

x_train = sc.fit_transform(x_train)
x_test = sc.fit_transform(x_test)
```

Training and Evaluating Machine Learning Model:

We can now train our classification model. We’ll be using a machine simple learning model called **SVC** (support vector classifier). Since the model is readily available in **sklearn**, the training process is quite easy and we can do it in few lines of code. First, we create an instance called **clf** and then use the fit function to train the model.

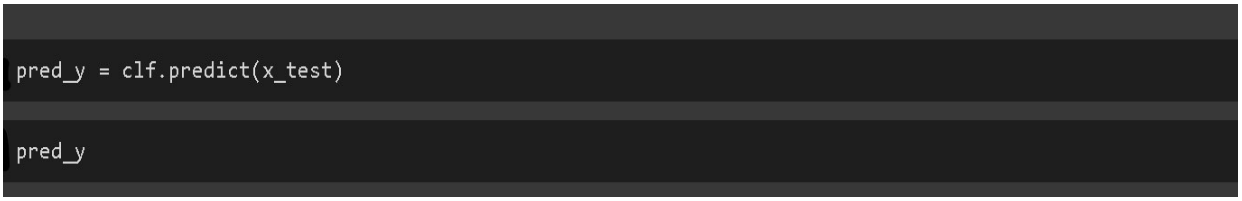


```
from sklearn.svm import SVC
clf = SVC(kernel="poly")

clf.fit(x_train, y_train)
```

Making Predictions with Model:

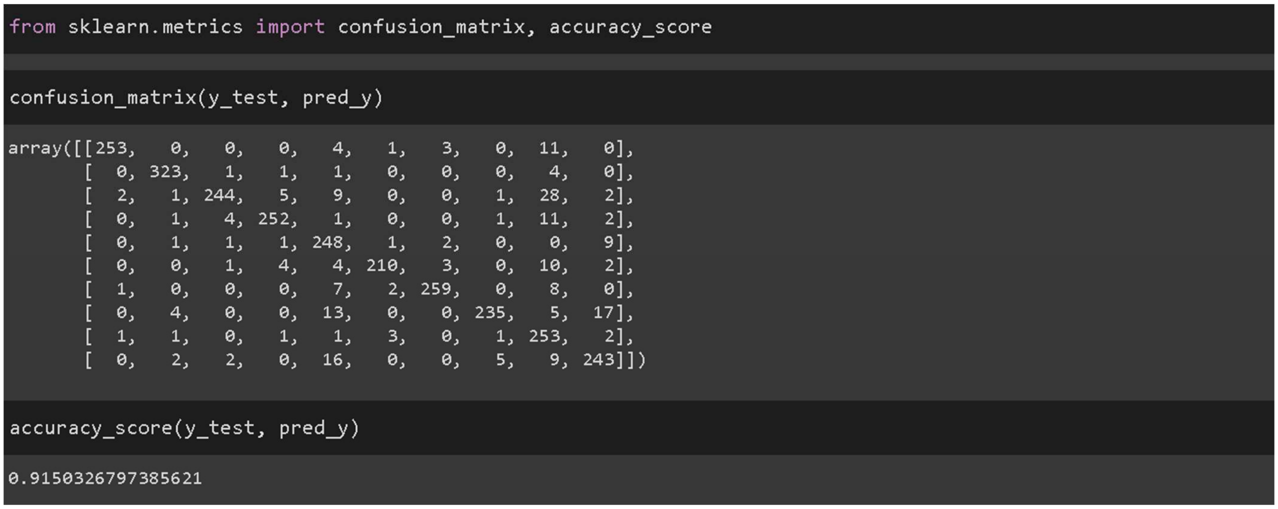
Now when the model is trained we will predict the digit. First, we create an instance called **pred_y** and then use the predict function to test the model.



```
pred_y = clf.predict(x_test)

pred_y
```

Now, to get confusion matrix and accuracy score, we will do following code:



```
from sklearn.metrics import confusion_matrix, accuracy_score
```

```
confusion_matrix(y_test, pred_y)
```

```
array([[253,   0,   0,   0,   4,   1,   3,   0,  11,   0],
       [  0, 323,   1,   1,   1,   0,   0,   0,   4,   0],
       [  2,   1, 244,   5,   9,   0,   0,   1,  28,   2],
       [  0,   1,   4, 252,   1,   0,   0,   1,  11,   2],
       [  0,   1,   1,   1, 248,   1,   2,   0,   0,   9],
       [  0,   0,   1,   4,   4, 210,   3,   0,  10,   2],
       [  1,   0,   0,   0,   7,   2, 259,   0,   8,   0],
       [  0,   4,   0,   0,  13,   0,   0, 235,   5,  17],
       [  1,   1,   0,   1,   1,   3,   0,   1, 253,   2],
       [  0,   2,   2,   0,  16,   0,   0,   5,   9, 243]])
```

```
accuracy_score(y_test, pred_y)
```

```
0.9150326797385621
```

BY,

DIVOLIKA BAJPAI